Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A compiler apparatus for collecting frequencies with which each process is executed in a program to be optimized and optimizing said program based on the collected frequencies, said apparatus comprising:

a loop process detection portion ~~for~~ to detect~~ing~~ a repeatedly executed loop process of said program;

a loop process frequency collection portion ~~for~~ to collect~~ing~~ loop process frequencies with which said loop process is executed in said program;

an in-loop process frequency collection portion ~~for~~ to collect~~ing~~ in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed;

an in-loop execution information generating portion ~~for~~ to, based on said loop process frequencies and said in-loop process frequencies, generate~~ing~~ in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed; and

an optimization portion ~~for~~ to optimize~~ing~~ said program based on said in-loop execution information generated by said in-loop execution information generating portion,

the in-loop process frequency collection portion further determining whether said loop process frequencies are higher than a predetermined reference frequency, and said in-loop process frequency collection portion determines number of times of execution of said each of a plurality of in-loop processes if said loop process frequencies are higher than a predetermined reference frequency.

2. (Original) The compiler apparatus according to claim 1, wherein said in-loop process frequency collection portion collects said in-loop process frequencies in the case where said loop process frequencies are higher than a predetermined frequency.

3. (Original) The compiler apparatus according to claim 1, wherein said in-loop execution information generating portion generates said in-loop execution information by multiplying said loop process frequencies by said in-loop process frequencies.

4. (Original) The compiler apparatus according to claim 1, wherein:

said loop process is an outer loop process including an inner loop process which is a further inside loop process;

said loop process detection portion further detects said inner loop process;

said loop process frequency collection portion collects the loop process frequencies with which said inner loop process is executed in said program based on said in-loop execution information;

said in-loop process frequency collection portion further collects the in-loop process frequencies of said inner loop process; and

said in-loop execution information generating portion generates the in-loop execution information on said inner loop process by multiplying the in-loop process frequencies in said inner loop process by said loop process frequencies of said inner loop process.

5. (Original) The compiler apparatus according to claim 1, wherein:

said loop process frequency collection portion stops a counter for determining the number of times of execution of said loop process when said program is executed a predetermined number of times so as to collect the number of times determined by the counter as said loop process frequencies; and

said in-loop process frequency collection portion stops the counter for determining the number of times of execution of each of said plurality of in-loop processes when a total of determined values of said plurality of in-loop processes becomes the predetermined number of times.

6. (Original) The compiler apparatus according to claim 1, further comprising:

a control flow graph generating portion for generating a control flow graph in which each of a plurality of instruction sequences in said program is generated as a node and an execution order of said plurality of instruction sequences is generated as a directed edge of said nodes;

a structure graph generating portion for, in said control flow graph, generating an outline structure graph in which a single loop node for showing said loop process in its entirety is generated instead of a collection of the nodes forming said loop process and an in-loop structure graph which is the control flow graph of the collection of the nodes forming said loop process; and

a counter insertion portion for, in each of said outline structure graph and said in-loop structure graph, inserting a counter into said program in order to count the number of times of execution of each execution path in the structure graphs, and wherein:

said loop process frequency collection portion generates as said loop process frequencies the numbers of times of execution of said loop node as against the numbers of times of execution of said program; and

said in-loop process frequency collection portion collects as said in-loop process frequencies the number of times of execution of each execution path in said in-loop structure graph as against the numbers of times of execution of said loop process.

7. (Original) The compiler apparatus according to claim 6, wherein:

in the case where said program is executed a predetermined number of times, said loop process frequency collection portion collects as the loop process frequencies the determined values of the counter inserted for counting the number of times of execution of the execution paths including said loop node; and

in the case where a total of the determined values of said plurality of in-loop processes becomes a predetermined number of times, said in-loop process frequency collection portion collects the in-loop process frequencies based on the determined values of the counter inserted for counting the number of times of execution of each execution path in said in-loop structure graph.

8. (Original) The compiler apparatus according to claim 6, wherein in the case where an insertion position in said program for inserting the counter for determining the number of times of execution of each execution path in said outline structure graph is the same as the position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-loop structure graph and then the counter of one, at the most, of said

outline structure graph and said in-loop structure graph is started, said counter insertion portion inserts into the insertion position the counter for determining the numbers of times of execution of the execution paths in both said outline structure graph and said in-loop structure graph.

9. (Original) The compiler apparatus according to claim 6, wherein:

in the case where an insertion position in said program for inserting the counter for determining the number of times of execution of each execution path in said outline structure graph is the same as the position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-loop structure graph and then the counter of one, at the most, of said outline structure graph and said in-loop structure graph is started, said counter insertion portion generates a plurality of determination processes for determining the number of times of execution of each execution path in each of said outline structure graph and said in-loop structure graph; and said in-loop process frequency collection portion inserts a jump instruction for moving the process to another portion into said insertion position and sets a jump destination of the jump instruction at one of said plurality of determination processes so as to determine the numbers of times of execution of the execution paths in both said outline structure graph and said in-loop structure graph.

10. (Original) The compiler apparatus according to claim 6, wherein:

said loop process is an outer loop process including an inner loop process which is a further inside loop process;

said loop process detection portion further detects said inner loop process;

in the control flow graph of said outer loop process, said structure graph generating portion generates as an in-outer loop structure graph a graph in which the single inner loop node is generated instead of a collection of the nodes forming said inner loop process and generates an in-inner loop structure graph which is the control flow graph of the collection of the nodes forming said inner loop process; and

said counter insertion portion further inserts the counter for determining the number of times of execution of each execution path in the in-inner loop structure graph;

said loop process frequency collection portion further collects the loop process frequencies with which said inner loop process is executed in said program based on said in-loop execution information;

said in-loop process frequency collection portion collects the frequencies of execution of each execution path in said in-inner loop structure graph as the in-loop process frequencies of said inner loop process as against the number of times of execution of said inner loop process; and

said in-loop execution information generating portion further generates the in-loop execution information on said inner loop process by multiplying the in-loop process frequencies in said inner loop process by the loop process frequencies of said inner loop process.

11. (Original) The compiler apparatus according to claim 10, wherein, in the case where an insertion position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-outer loop structure graph is the same as the position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-inner loop structure graph and then the counter of one, at the most, of said in-outer loop structure graph and said in-inner loop structure graph is started, said counter insertion portion inserts into the insertion position the counter for determining the numbers of times of execution of the execution paths in both said in-outer loop structure graph and said in-inner loop structure graph.

12. (Original) The compiler apparatus according to claim 10, wherein:

in the case where an insertion position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-outer loop structure graph is the same as the position in said program for inserting the counter for determining the number of times of execution of each execution path in said in-inner loop structure graph and then the counter of one, at the most, of said in-outer loop structure graph and said in-inner loop structure graph is started, said counter insertion portion generates a plurality of determination processes for determining the number of times of execution of each execution path in each of said in-outer loop structure graph and said in-inner loop structure graph; and said in-loop process frequency collection portion inserts a jump instruction for moving the process to another portion into said insertion position and sets a jump destination of the jump instruction at one of said plurality of determination processes so as to determine the number of times of execution of the execution paths in both said in-outer loop structure graph and said in-inner loop structure graph.

13. (Currently Amended) A compiler program for causing a computer to function as a compiler apparatus for collecting frequencies with which each process is executed in a program to be optimized and optimizing said program based on the collected frequencies, said program causing said computer to function as:

a loop process detection portion ~~for~~ to detect~~ing~~ a repeatedly executed loop process of said program;

a loop process frequency collection portion ~~for~~ to collect~~ing~~ loop process frequencies with which said loop process is executed in said program;

an in-loop process frequency collection portion ~~for~~ to collect~~ing~~ in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed;

an in-loop execution information generating portion ~~for~~ to, based on said loop process frequencies and said in-loop process frequencies, generate~~ing~~ in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed; and

an optimization portion ~~for~~ to optimize~~ing~~ said program based on said in-loop execution information generated by said in-loop execution information generating portion,

the in-loop process frequency collection portion further determining whether said loop process frequencies are higher than a predetermined reference frequency, and said in-loop process frequency collection portion determines number of times of execution of said each of a plurality of in-loop processes if said loop process frequencies are higher than a predetermined reference frequency.

14. (Currently Amended) ~~The~~ A record medium having the compiler program according to claim 13 recorded thereon.

15. (Currently Amended) A compilation method for collecting frequencies with which each process is executed in a program to be optimized and optimizing said program based on the collected frequencies, said method having:

a loop process detection step of detecting a repeatedly executed loop process of said program;

a loop process frequency collection step of collecting loop process frequencies with which said loop process is executed in said program;

an in-loop process frequency collection step of collecting in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed;

an in-loop execution information generating step of, based on said loop process frequencies and said in-loop process frequencies, generating in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed; and

an optimization step of optimizing said program based on said in-loop execution information generated by said in-loop execution information generating portion,

the in-loop executing information generating step further determining whether said loop process frequencies are higher than a predetermined reference frequency, and determining number of times of execution of said each of a plurality of in-loop processes.

16. (Currently Amended) A runtime information generating apparatus for collecting frequencies with which each process is executed in a program to be optimized, said apparatus having:

a loop process detection portion ~~for~~ to detect~~ing~~ a repeatedly executed loop process of said program;

a loop process frequency collection portion ~~for~~ to collect~~ing~~ loop process frequencies with which said loop process is executed in said program;

an in-loop process frequency collection portion ~~for~~ to collect~~ing~~ in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed;

an in-loop execution information generating portion ~~for~~ to, based on said loop process frequencies and said in-loop process frequencies, generate~~ing~~ in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed, and optimize~~ing~~ said program based on said in-loop execution information generated by said in-loop execution information generating portion,

the in-loop process frequency collection portion further determining whether said loop process frequencies are higher than a predetermined reference frequency, and said in-loop process frequency collection portion determines number of times of execution of said each of a plurality of in-loop processes if said loop process frequencies are higher than a predetermined reference frequency.

17. (Currently Amended) A runtime information generating program for causing a computer to function as a runtime information generating apparatus for collecting frequencies with which each process is executed in a program to be optimized, said program causing said computer to function as:

a loop process detection portion ~~for~~ to detecte~~ing~~ a repeatedly executed loop process of said program; a loop process frequency collection portion for collecting loop process frequencies with which said loop process is executed in said program;

an in-loop process frequency collection portion ~~for~~ to collect~~ing~~ in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed; and

an in-loop execution information generating portion ~~for~~ to, based on said loop process frequencies and said in-loop process frequencies, generate~~ing~~ in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed, and causing said program to be optimized based on said in-loop execution information generated by said in-loop execution information generating portion,

the in-loop process frequency collection portion further determining whether said loop process frequencies are higher than a predetermined reference frequency, and said in-loop process frequency collection portion determines number of times of execution of said each of a plurality of in-loop processes if said loop process frequencies are higher than a predetermined reference frequency.

18. (Original) A record medium having a runtime information generating program according to claim 17 recorded thereon.

19. (Original) A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing collection of frequencies with which each process is executed in a program to be optimized, the computer readable program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 1.

20. (Original) An article of manufacture comprising a computer usable medium having computer readable program code means embodied therein for causing collection of frequencies with which each process is executed in a program to be optimized, the computer readable program code means in said article of manufacture comprising computer readable program code means for causing a computer to effect the steps of claim 15.

21. (Original) A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for collecting frequencies with which each process is executed in a program to be optimized, said method steps comprising the steps of claim 15.

22. (Original) A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing collection of frequencies with which each process is executed in a program to be optimized, the computer readable program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 16.

23. (Currently Amended) A runtime information generating method comprising:

collecting frequencies with which each process is executed in a program to be optimized, said step of collecting frequencies comprising:

detecting a repeatedly executed loop process of said program;

collecting loop process frequencies with which said loop process is executed in said program;

collecting in-loop process frequencies with which, as against the number of times of execution of said loop process, each of a plurality of in-loop processes included in said loop process is executed;

based on said loop process frequencies and said in-loop process frequencies, generating in-loop execution information indicating the frequencies with which each of said plurality of in-loop processes is executed in the case where said program is executed, and optimizing said program based on said in-loop execution information generated by said in-loop execution information generating portion,

said generating further determining whether said loop process frequencies are higher than a predetermined reference frequency, and determining number of times of execution of said each of a plurality of in-loop processes if said loop process frequencies are higher than a predetermined reference frequency.

24. (Original) An article of manufacture comprising a computer usable medium having computer readable program code means embodied therein for causing runtime information generation, the computer readable program code means in said article of manufacture comprising computer readable program code means for causing a computer to effect the steps of claim 23.

25. (Original) A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for runtime information generation, said method steps comprising the steps of claim 15.